FB401 入门教程之组网灯控

一、关于本文档

本文档介绍了开发一个简单的组网灯控方案的方法，帮助开发者了解通过蓝牙网络实现设备控制的基本思路。

二、前提条件

在动手之前，请确认如下条件已经满足：

1. 您已经阅读了 ‘快速上手’ 教程并跑通例程

如果没有，请务必先阅读 ‘快速上手’。

三、范例工程

本教程基于 SDK 中的 Examples\Light 范例。

四、设计目标

我们首先确定一个简单目标：实现一个两通道的简单灯控，即一个控制模块控制两个 LED 灯。第一步我们不考虑 PWM 调光，只考虑开关。两通道灯控通常用来实现冷光、暖光的切换。

五、基本思路

一个通道对应一个 GPIO 输出，我们要做两通道，所以会用到两个 GPIO 输出。这里我们选择使用 PA4 和 PA6 对应着两个通道，如下：

通道	GPIO
1	PA4
2	PA6

当然你也可以选择用其他的 GPIO。

然后我们来构思一下程序的架构。在 FB401 平台上，我们不需要操心系统的初始化，蓝牙协议栈的运行之类，我们只需要考虑这个应用需要什么特定的逻辑。对于一个灯控，我们需要考虑的就是两件事：

GPIO 的初始化

GPIO 的拉高、拉低操作

首先我们来想一下在什么时间做上述的动作，不难想到：

时机	动作
系统初始化完成	GPIO 初始化
收到控制命令	GPIO 拉高、拉低操作

在 FB401 平台，系统初始化完成是一个隶属于 SYS 模块的一个事件，这个事件 Hello World 例程就会用到，我们找出来代码看一下：

```
// SYS 模块事件处理程序
void SYS_EventHandler(uint32_t event, void *params)
{
    switch (event)
    {
    case SYS_EVENT_READY:
        printf("Hello World\r\n");
    }
}
```

很明显，我们只要把红色部分给替换成 GPIO 初始化的代码，即可达到在系统初始化完成后，初始化 GPIO 的目的。

GPIO 初始化的接口是这样的：

```
GPIO_SetPinMode(port, pin, mode);
```

这里，port 指的是 PA,PB,PC,PD，在代码里面表示为 GPIO_PA, GPIO_PB 以此类推。

Pin 指的是从 0 到 7 的编码，代码里面是 GPIO_PIN_0， GPIO_PIN_1，以此类推

Mode 指的是要配置的模式，比如输入，输出，我们要配置的是输出模式 GPIO_PIN_MODE_OUTPUT。

关于接口的完整说明请查阅参考手册。

那么，现在我们打算把 PA4 和 PA6 给初始化成输出模式。代码就是这样的：

```
GPIO_SetPinMode(CH1_PORT, CH1_PIN, GPIO_PIN_MODE_OUTPUT);
GPIO_SetPinMode(CH2_PORT, CH2_PIN, GPIO_PIN_MODE_OUTPUT);
```

注意这里我们定义了几个宏以提高代码的可读性，他们是：

```
#define CH1_PORT GPIO_PA
#define CH1_PIN GPIO_PIN_4

#define CH2_PORT GPIO_PA
#define CH2_PIN GPIO_PIN_6
```

他们的含义显而易见。

初始化完成后，我们还希望有一个默认状态，比如我们希望两个通道默认都是高电平。这就要用到 GPIO 的写操作，这个接口是这样的：

```
GPIO_WritePin(port, pin, value);
```

这个接口和 SetPinMode 很像。Port 和 pin 就不用解释了。Value 的意思是要写入的值，有 0 和 1 两个，对应低电平和高电平。

所以我们要加的两行代码是：

```
GPIO_WritePin(CH1_PORT, CH1_PIN, 1);
GPIO_WritePin(CH2_PORT, CH2_PIN, 1);
```

现在初始化代码写好了。这时候如果我们运行工程，就会看到两个通道拉高。下一步是在接受到指令的时候进行反应。这时我们就要构思一个指令格式。对于这个应用，我们构思一个非常简单的格式：用一个字节的数据，其中最低两个 bit 对应两个通道的状态：

通道	BIT 掩码
1	0x01
2	0x02

那么我们如何接收数据呢？FB401 平台会管理好底层通信，我们只需要对收到的数据进行处理即可。直接上相关代码，一看便知：

```
// BLE 模块事件处理程序
void BLE_EventHandler(ble_event_t event, void *params)
{
```

```
    switch (event)
    {
    case BLE_EVENT_DATA_RX:
    {
        ble_data_rx_event_params_t *ep =
(ble_data_rx_event_params_t *)params;
        uint8_t cmd = ep->data[0];

        // printf("CMD: %02X\n", cmd);

        // 我们这个简单的灯控应用,只有一个字节的指令
        // 用这个字节的最低两个 bit 分别控制两个 GPIO PIN 的高低

        // bit 0x01 是通道 1
        if (cmd & 0x01)
        {
            // 通道 1 开
            GPIO_WritePin(CH1_PORT, CH1_PIN, 1);
        }
        else
        {
            // 通道 1 关
            GPIO_WritePin(CH1_PORT, CH1_PIN, 0);
        }

        // bit 0x02 是通道 2
        if (cmd & 0x02)
        {
            // 通道 2 开
            GPIO_WritePin(CH2_PORT, CH2_PIN, 1);
        }
        else
        {
            // 通道 2 关
            GPIO_WritePin(CH2_PORT, CH2_PIN, 0);
        }

        break;
    }
    }
}
```

这段代码里面，我们定义了一个事件处理函数，处理了一个接收数据的事件，然后根据数据的内容控制两个 GPIO 的电平。

然后怎么测呢？我们可以用微信打开 FB401 的测试程序（蓝牙演示工具），连上我们的硬件，然后发送命令。这部分具体内容可以参考 ‘快速上手’。

值得一提的是，FB401 默认内建了组网（Mesh）功能，如果你烧录多个模块，你就可以连接一个节点但是控制所有的节点。关于 Mesh 的更多内容，也请看参考手册。

六、小结

完整的工程在 Examples\Light。但我们也建议读者亲自基于 Hello World 工程一步一步添加代码建立起来自己的灯控工程，这样就可以有更加深刻的体会。

您可以看出基于 FB401 平台开发应用是一个非常简单直接的过程。掌握了灯控开发的方法后，不知道您是不是想开发更多有趣、有用的智能硬件了呢？